THIS OPINION WAS NOT WRITTEN FOR PUBLICATION

The opinion in support of the decision being entered today (1) was not written for publication in a law journal and (2) is not binding precedent of the Board.

Paper No. 12

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte KARL M. GUTTAG,
SYDNEY W. POLAND,
and KEITH BALMER

Appeal No. 96-0908
Application 08/160,118[1]

ON BRIEF

Before HAIRSTON, JERRY SMITH, and BARRETT, Administrative Patent Judges.

BARRETT, Administrative Patent Judge.

DECISION ON APPEAL

[1] Application for patent filed November 30, 1993, entitled "Memory Store From A Register Pair Conditional Upon A Selected Status Bit."

This is a decision on appeal under 35 U.S.C. § 134 from the final rejection of claims 1-43.

We affirm-in-part.

BACKGROUND

The disclosed invention is directed to a data processing apparatus wherein the contents of one of two registers is stored to memory depending upon the state of a status bit in response to a register pair conditional store instruction.

Claim 1 is reproduced below.

> 1. A data processing apparatus comprising:
>
> a memory storing data at addressable memory locations;
>
> an addressing circuit generating memory addresses for data accesses to said memory;
>
> a data circuit including
>
> > a plurality of data registers, each storing a predetermined number of data bits,
> >
> > a status register storing at least one type of status bit, and
> >
> > an arithmetic logic unit having operand inputs and an output coupled to said plurality of data registers; and
>
> an instruction logic circuit connected to said addressing circuit and said data circuit, said instruction logic circuit controlling said addressing circuit and said data circuit in response to a received

instruction, said instruction logic circuit controlling said addressing circuit and said data circuit to store said predetermined number of data bits stored in a first data register into a specified address in said memory if a status bit selected from said at least one type of status bit has a first state and to store said predetermined number of data bits stored in a second data register associated with said first data register into said specified address in said memory if a status bit selected from said at least one type of status bit has a second state in response to a register pair conditional store instruction.

The examiner relies on the following prior art references:

Auslander et al. (Auslander)	4,589,087	May 13, 1986
Diefendorff et a. (Diefendorff)	5,268,995	December 7, 1993 (filed November 21, 1990)
Kawata	5,274,777	December 28, 1993 (filed March 29, 1991)

We refer to the Final Rejection (Paper No. 5) (pages referred to as "FR__") and the Examiner's Answer (Paper No. 11) (pages referred to as "EA__") for a statement of the examiner's position and to the Appeal Brief (Paper No. 10) (pages referred to as "Br__") for a statement of appellants' arguments thereagainst. The outstanding rejections are:

Claims 1-43 stand rejected under 35 U.S.C. § 112, second paragraph. The examiner states that in claims 1 and 13, with respect to "'a register pair conditional store instruction', it is not clear whether it refers to a special purpose instruction or a conventional conditional instruction" (FR2).

Claims 1-43 stand rejected under 35 U.S.C. § 103 as being unpatentable over Auslander and Diefendorff.

Claims 1-43 stand rejected under 35 U.S.C. § 103 as being unpatentable over Kawata and Diefendorff.

OPINION

35 U.S.C. § 112, second paragraph

We agree with appellants that the term "register pair conditional store instruction" is definite. Claims 1 and 13 describe the action performed "in response to" the instruction and the "register pair conditional store instruction" is descriptive of the action. It is not understood how the kind of instruction has anything to do with definiteness. The rejection of claims 1-43 under § 112, second paragraph, is reversed.

35 U.S.C. § 103

Grouping of claims

Appellants identify the following groups of claims (Br3):

(1) claims 1-3, 13-15, 26-39 [sic, 25-39];
(2) claims 4 and 16;
(3) claims 5 and 17;
(4) claims 6 and 18;
(5) claims 7 and 19;
(6) claims 40 and 42;
(7) claims 8 and 20;
(8) claims 9 and 21;
(9) claims 10 and 22;
(10) claims 41 and 42 [sic, 43];
(11) claims 11 and 23;
(12) claims 12 and 24.

The examiner regroups the claims (EA2), but we will follow appellants' groupings.

Level of ordinary skill in the art

The references are considered to be representative of the level of ordinary skill in the art. See In re Oelrich, 579 F.2d 86, 91, 198 USPQ 210, 214 (CCPA 1978) ("the PTO usually must evaluate both the scope and content of the prior art and the level of ordinary skill solely on the cold words of the literature"); In re GPAC, Inc., 57 F.3d 1573, 1579, 35 USPQ2d 1116, 1121 (Fed. Cir. 1995) (the Board did not err in adopting the approach that the level of skill in the art was best determined by the references of record). Obviousness is determined through the eyes of one of ordinary skill in the art and that person must be presumed to know something about the art apart from what the references expressly disclose. See In re Jacoby, 309 F.2d 513, 516, 135 USPQ 317, 319 (CCPA 1962); In re Oetiker, 977 F.2d 1443, 1447-48, 24 USPQ2d 1443, 1446-47 (Fed. Cir. 1992) (Nies, C.J., concurring).

The "art to which the subject matter pertains" in § 103 in this case is the art of computer hardware design and computer instruction set design. We find that one of ordinary skill in this art has a very high level of education, training, and experience due to the complex nature of the subject matter. One of ordinary skill in this art is presumed to have knowledge of, at least, commercial computer designs and instruction sets.

Content of the prior art

Diefendorff

Diefendorff discloses a load/store machine for executing graphic Z-compare and pixel merge instructions. Diefendorff discloses that, in response to a Z-compare instruction, a first source operand which contains two 32-bit fields (S_z0 and S_z1) representing Z-values for two new pixels (S0, S1) is retrieved from register 52 and a second source operand which contains two 32-bit fields (D_Z0 and D_Z1) representing Z-values for two currently stored pixels (D0, D1) is retrieved from register 54 (figures 3 and 4; col. 8, lines 3-10). The operands are transferred to the graphics adder unit 30, which performs two 32-bit Z-value comparisons to determine which value is greater and the two "greater than" results are encoded as two bits in the mask register 56. The mask bit values select which portions of the registers 57 and 58 are stored in register 59 by the multiplexers 60-67 in figure 5 in response to a merge instruction. "Thus, the final result operand is stored in register 59 in the register file 34." Col. 9, lines 64-65.

Diefendorff further discloses (col. 11, lines 22-31):

> The mask value produced by the Z-compare instructions may be used by a pixel merge instruction which places the pixel result operand into the register file 34 for later storage by a STORE instruction executed by the load/store units 25. Alteratively [sic], a CONDITIONAL-STORE instruction executed by the load/store units 25 may directly use the mask value to independently enable the transfer of each byte in the result operand from the register file 34 to memory 50 or a video frame buffer (not shown) via the data bus 48. [Emphasis added.]

Auslander

Auslander discloses a primitive reduced instruction set machine wherein every primitive instruction takes exactly one machine cycle, except for accessing storage. The system architecture has a condition code generating means for generating condition bits in accordance with the output of the arithmetic logic unit (ALU) and an expanded condition register (claim 1). Auslander discloses that a number of condition bits (status bits) can be set by the ALU, including, as shown in Table 1(a), a Carry (CA) bit, an Overflow (OV) bit, a Compares Equal, Zero Value (EQ) bit, and a Logical Greater Than (LG) bit. Certain condition bits are not altered by certain instructions, e.g., the Carry bit (CA) "is not altered by the compare instruction" (col. 9, lines 29-30) and the divide step (DVS) instruction sets the Carry bit (CA) and Overflow bit (OV) but does not change the other condition bits (col. 13, lines 6-11).

<u>Kawata</u>

Kawata discloses a digital data processor which executes a conditional instruction within a single machine cycle, which is used in sorting pieces of data. The processor has an ALU. "The ALU subtracts the input data from the first and second buses for comparison and sets the sign bit of the conditional code ('S' indicated at the register 15 in FIG. 1) to 0. This sign bit indicates that the result obtained by subtracting the content of the second bus from the content of the first bus is a positive value." (Col. 4, lines 52-57.)

(1) Claims 1-3, 13-15, and 25-39

Appellants argue that the rejection fails to comply with the requirements of 37 CFR § 1.106(b) and fails to give appellants fair notice of the portions of the references relied on (Br4-5). The examiner's failure to comply with Patent and Trademark Office rules is a petitionable matter. The Board's jurisdiction is limited to those matters involving the rejection of claims. See In re Hengehold, 440 F.2d 1395, 1404, 169 USPQ 473, 480 (CCPA 1971). The examiner's reasoning in the Final Rejection could have been more detailed, but we understand what was intended. The references are not lengthy and the pertinence of each is apparent. The references as a whole are relied on, not just the portions mentioned by the examiner.

Diefendorff discloses (col. 11, lines 26-30): "Alteratively [sic], a CONDITIONAL-STORE instruction executed by the load/store units 25 may directly use the mask value to independently enable the transfer of each byte in the result operand from the register file 34 to memory 50." Thus, Diefendorff discloses a conditional store to memory of the result operand based on the mask (status) value, which implies that the bytes are taken from registers 57, 58 in the general register file 34 rather than being stored first in register 59 in the register file 34. Even if the result operand were stored first in register 59 in the general register file 34, then stored to memory, claim 1 does not exclude an intermediate storage as part of the action produced by the CONDITIONAL-STORE instruction. Diefendorff alone is sufficient to establish a <u>prima facie</u> case of obviousness.

Appellants argue (Br5): "First, a single selected status bit controls the storage of all of the bits of the registers. This recitation is not shown in Auslander et al. Diefendorff et al shows plural status bits control whether differing parts of registers 57 and 58 are stored in register 59." We do not find a comparable argument for the Kawata rejection.

Diefendorff operates on multiple fields within the 64-bit registers. Each field comparison between Z-values for new pixels in register 52 and currently stored pixels in register 54 produces one bit in the result (mask) register 56 (which can also be called a status register or flag register). Figure 4 shows a 32-bit field size which produces the two leftmost result bits in register 56 which are used in the merge operation in figure 5. Figure 6 shows an 8-bit field size which produces eight result bits. One of ordinary skill would have appreciated from these two examples that the field size can be the width of the whole register where the comparison produces one status bit. Moreover, the claim language, as broadly interpreted, does not distinguish over Diefendorff. Claim 1 recites "data registers, each storing a predetermined number of data bits." The "predetermined number of data bits" does not have to be the total number of bits in the register, but could be, for example, the 32-bit field in figure 3. Thus, this argument of appellants is not persuasive.

Appellants argue in the Auslander rejection (Br5): "Second, Diefendorff et al teaches storing the resultant of his pixel merge instruction in a register in general register file 34 and not at an addressed location within memory as claimed." Similarly, appellants argue in the Kawata rejection (Br14): "First, both Kawata and Diefendorff et al teach storing resultant data in a general register file rather than an addressed location in memory as claimed."

As previously discussed, Diefendorff discloses an alternative embodiment in which a CONDITIONAL-STORE instruction is executed by the load/store units 25 to directly use the mask value to transfer each byte in the result operand from the register file 34 to memory 50, instead of storing the result operand in a register and then using a STORE instruction. Since the result operand is taken from one of two operand registers, Diefendorff's CONDITIONAL-STORE instruction is a two register conditional store instruction as claimed. Even if the result operand were stored first in register 59 in the general register file 34, then stored to memory, claim 1 does not exclude an intermediate storage as part of the action produced by the CONDITIONAL-STORE instruction. Appellants do not address the CONDITIONAL-STORE teaching of Diefendorff even though they quote from and argue the teachings at column 11, lines 15-17 of Diefendorff (Br6 and Br14), which is in the preceding paragraph.

The examiner states that the claims do not require direct storing of selected bits to memory because the claims do not preclude additional intervening steps (EA7). It is true that claim 1's recitation of "said instruction logic circuit controlling said addressing circuit and said data circuit to store said predetermined number of data bits stored in a first data register into a specified address in said memory if a status bit . . . has a first state" does not exclude an intermediate step of storing the contents of the first register in another register. However, because the store must be "in response to a register pair conditional store instruction," which limitation the examiner apparently overlooks, the store must be performed with one instruction. Storing in an intermediate register requires an additional STORE instruction to move the results to memory (Diefendorff, col. 11, lines 22-26). We agree with the examiner's reasoning that one of ordinary skill in the art would have recognized that if the whole register was stored the pixel merge operation would be eliminated and the result could be sent directly to memory (EA7), but it is not necessary to rely on this reasoning since Diefendorff expressly discloses a direct store to memory in response to the CONDITIONAL-STORE instruction.

Appellants argue (Br6):

> Third, claims 1 and 13 specifically recite that the second data register is "associated with said first data register." Note the [sic] both claims 1 and 13 make clear that the first and second data registers are within the equivalent of the general register files 34 of Diefendorff et al. Registers 57 and 58 of Diefendorff et al are not registers in general register files 34 nor are they "associated" with each other in the sense claimed.

Similar arguments are made with respect to the rejection over Kawata (Br15-16).

These arguments are not commensurate in scope with the claims. Claims 1 and 13 do not recite where the data registers are located. However, if the claims did recite that they were in the general register files, Auslander suggests that the first and second source operands are retrieved from registers 52 and 54 in the general register file 34 and sent to the graphics adder unit 30 (col. 7, line 66, to col. 8, line 14). Kawata also teaches that the operands can be read from general registers 10 (figure 1), which suggests the obviousness of such a limitation. Claims 1 and 13 do not define what is meant by "associated with." "Associated with" can be broadly interpreted as associated in the sense that the two registers are identified, expressly or impliedly, as part of the instruction. The specific associations by consecutive register numbers (e.g., claim 8) or the second register number having a register number one less than the first register number (e.g., claim 9) are not found in claims 1 and 13.

For the reasons discussed above, we sustain the rejections of claims 1-3, 13-15, and 25-39.

<u>(2) Claims 4 and 16</u>
<u>(3) Claims 5 and 17</u>
<u>(4) Claims 6 and 18</u>

Claims 4-6 and 16-18 are directed to setting a particular status bit (flag bit or condition bit) corresponding to the output of the arithmetic logic unit (ALU). The claims do not require that the status bit is the status bit in claim 1 that controls the conditional store operation. The status bit could be just one of the status bits in the status register set by an ordinary instruction. Diefendorff discloses setting a "greater than" condition bit (status bit) from a comparison of two operands using unsigned arithmetic (col. 8, lines 63-68). Since Diefendorff is directed to a Z-compare to determine which pixel is less, it does not disclose other kinds of comparisons. Auslander discloses that a number of condition bits (status bits) can be set by the ALU, including, as shown in Table 1(a), a Carry (CA) bit as recited in claims 4 and 16, an Overflow (OV) bit as recited in claims 5 and 17, a Compares Equal, Zero Value (EQ) bit as recited in claims 6 and 18, and a Logical Greater Than (LG) bit, which one skilled in the art would recognize as corresponding to Diefendorff's condition bit.

The examiner took official notice that "[d]ifferent types of condition status bits and combination[s] of these bits have also been used in prior arts for controlling the execution of instructions" (FR6). We agree with the finding that one of ordinary skill in the art would have known that the claimed carry status bit (claims 4 and 16), overflow status bit (claim 5), and zero status bit (claims 6 and 18) are well known and conventional status (or flag or condition) bits in the computer art. See Microprocessors (Intel Corp. 1989), pages 4-11 to 4-14 (copy attached) (showing flags register for Intel 486 microprocessor).[2] One skilled in the art would have

[2] This reference is cited to show what was known by those of ordinary skill in the art. See In re Boon, 439 F.2d 724, 727-28, 169 USPQ 231, 234 (CCPA 1971) (standard work cited to support an officially noticed fact which plays a minor role does not raise a new ground of rejection).

been motivated to provide the claimed status bits for the results of ALU operations in a computer in view of Auslander and common knowledge in the art. The rejections of claims 4-6 and 16-18 over Auslander and Diefendorff and Kawata and Diefendorff are sustained.

Appellants argue that Diefendorff discloses a graphics merge instruction in the context of a Z-buffering system and "Diefendorff et al provides no motivation to employ another type of status determination because his disclosure is narrowly directed to only a single problem" (Br8, lines 4-6, and Br17, lines 27-29; similar statements at Br8, lines 30-33, and Br9, lines 21-23, Br18, lines 25-27, Br19, lines 24-26). The status bits have no claimed relationship to the conditional store instruction. One of ordinary skill in the pertinent art would have been motivated to provide the status bits for other instructions since the status bits were well known and conventional in the computer art as shown by Auslander. Assuming, arguendo, that there was a relationship between the status bits and the conditional store instruction, one of ordinary skill in the art is presumed to have had sufficient knowledge to generalize the two register conditional store operation of Diefendorff based on a "greater than" condition to other conditions such as "carry," "overflow," and "zero."

Appellants argue that "Auslander et al also fails to teach or suggest any usefulness of employing a status bit other than that disclosed in Diefendorff et al" (Br8, lines 11-13; Br9, lines 2-4 and lines 28-30). We note that claims 4-6 and 16-18 do not recite using the status bit as the status bit for the conditional store, but only require that a status bit be set. Auslander teaches all of the disclosed status bits.

Appellants argue that "[t]he only status bit mentioned in Kawata is a sign status bit" (Br17). We agree that Kawata discloses that the ALU set a sign bit S in the conditional code register (col. 4, lines 52-57) as the only status bit. The examiner does not address this deficiency in Kawata. However, since the status bits were conventional in the prior art, Diefendorff alone is sufficient to meet these claims.

(5) Claims 7 and 19
(6) Claims 40 and 42

Claims 7 and 19 require that the "status register stores a plurality of different types of status bits." Auslander discloses a plurality of different types of status bits. Kawata discloses only a sign bit and does not disclose a plurality of different types of status bits; however, as discussed supra, it was well known in the prior art to have a status register for a plurality of different types of status bits.

Claims 7 and 19 further require that "said register pair conditional store instruction includes a plurality of bits designating whether particular ones of said plurality of different types of status bits are protected from being set corresponding to said output of said arithmetic logic unit." Appellants argue (Br10): "Note that the quoted claim language clearly indicates that it is the particular bits of the current instruction that determines the status bit protection and not the instruction type as suggested in the rejection. The Appellants respectfully submit that Auslander et al does [sic, do not] teach or suggest that the instruction includes bits that determine whether a status bit is protected from change" (Br10). A similar argument is made with respect to the combination of Kawata and Diefendorff (Br20).

The examiner notes that the instruction operation in Auslander "sets one or more condition bits but not all condition bits" (EA11) and "[t]hus, the unaffected bits are deemed to be protected" (EA11). This does not address why one of ordinary skill in the art of instruction set design would have been motivated to provide masking bits <u>in the instruction</u> as claimed. The examiner could have provided examples of other instructions that had masking bits, but has not done so. Auslander discloses that certain instructions set some status bits, but leave others unchanged; e.g., the Divide Step (DVS) instruction changes the Carry (CA) and Overflow (OV) condition bits, but other condition bits are unchanged (col. 13, lines 6-11). The unchanged bits might be considered to be "protected." However, Auslander says nothing about the instruction including bits indicating which bits are to be protected. Kawata discloses only a sign status bit and does not describe changing only some status bits. The examiner has failed to establish a *prima facie* case of obviousness. The rejections of claim 7 and 19, and their dependent claims 40 and 42, are reversed.

(7) Claims 8 and 20
(8) Claims 9 and 21

Claims 8 and 20 recite that the data registers are accessed via consecutive register numbers, where the register pair conditional store instruction designates the first register by register number and the second register is accessed by a consecutive register number. While the intent of the claims is that the second register does not have to be designated in the instruction, the claims do not preclude the instruction designating the second register.

Diefendorff does not specify the format of the CONDITIONAL-STORE instruction. The examiner states (FR6): "Finally, a specific register numbering, i.e. in claims 8,9,20,21, not explicitly taught by the cited arts would have been an obvious engineering design choice to a skilled artisan." Design choice is not a substitute for evidence or obviousness reasoning based on what was known to those of ordinary skill in the art. The examiner could have provided examples of other types of instructions that used consecutive registers, explicitly or implicitly, but has not done so. The examiner has failed to establish a prima facie case of obviousness. The rejections of claim 8 and 20, and dependent claims 9 and 21, over Auslander and Diefendorff and Kawata and Diefendorff are reversed.

(9) Claims 10 and 22

Claims 10 and 22 recite a plurality of different types of status bits, which limitation we held to have been obvious in connection with the analysis of claim 7.

Claims 10 and 22 further recite that the register pair conditional store instruction designates a particular one of the plurality of different types of status bits for controlling which register is stored in memory. Appellants argue that "neither Auslander et al nor Diefendorff et al teach [sic] or suggest the particular conditional operation claimed" (Br12). A similar argument is made with respect to the rejection over the combination of Kawata and Diefendorff (Br22).

In our opinion, one of ordinary skill in the art had sufficient knowledge to generalize the two register conditional store instruction of Diefendorff, which is based on a "greater than" condition, to other status conditions, such as "carry," "overflow," and "zero." The status bit to be used with the register pair conditional store instruction can be designated by the instruction type and is not required to be a mask bit in the instruction. The rejections of claims 10 and 22 over Auslander and Diefendorff and Kawata and Diefendorff are sustained.

(10) Claims 41 and 43

Claims 41 and 43 depend on claims 10 and 22, respectively, and recite at least two status bits from the group of a negative status bit, a carry status bit, an overflow status bit, and a zero status bit.

As discussed with respect to the rejection of claims 4-6 and 16-18, we agree with the examiner's finding that one of ordinary skill in the art would have known that the claimed negative status bit, carry status bit, overflow status bit, and zero status bit are conventional status (or flag or condition) bits.
One skilled in the art would have been motivated to provide two or more of the claimed status bits for the results of ALU operations in view of Auslander and common knowledge in the

art. As discussed with respect to the rejection of claims 10 and 22, it would have been obvious to one having ordinary skill in this art to make the register pair conditional store instruction of Diefendorff dependent on different status bits. Appellants' argument that "Diefendorff et al fails [sic] to teach or suggest that his pixel merge instruction may be conditioned upon any two of these four named type of status bits as claimed" (Br23) fails to account for the knowledge of those skilled in the art. The rejections of claims 41 and 43 over Auslander and Diefendorff and Kawata and Diefendorff are sustained.

(11) Claims 11 and 23
(12) Claims 12 and 24

Claims 11 and 23 recite the operation of a "register pair conditional write instruction" which conditionally supplies the content of one of two registers to the first input of the ALU based upon the state of a status bit.

Appellants state that "[t]he rejection has pointed out no part of Auslander et al or Diefendorff et al that makes obvious selection of the input to an arithmetic logic unit from between two registers based upon a status bit" (Br12).

The same argument is made with respect to the combination of Kawata and Diefendorff (Br22). The examiner's response is (EA12): "The examiner submits that Auslander clearly teaches that multiple conditional operations can be executed in sequence, each operation utilizes different conditional bits." This reasoning does not address the claim limitations to the write instruction. None of the references disclose or suggest a conditional write instruction as claimed. The examiner has failed to establish a _prima facie_ case of obviousness over either Auslander or Kawata in view of Diefendorff. The rejections of claim 11 and 23, and their dependent claims 12 and 24, over Auslander and Diefendorff and Kawata and Diefendorff are reversed.

CONCLUSION

The rejection of claims 1-43 under 35 U.S.C. § 112, second paragraph, is reversed.

The rejections of claims 1-6, 10, 13-18, 22, 25-39, 41, and 43 are sustained.

The rejections of claims 7-9, 11, 12, 19-21, 23, 24, 40, and 42 are reversed.

No time period for taking any subsequent action in connection with this appeal may be extended under 37 CFR § 1.136(a).

AFFIRMED-IN-PART

KENNETH W. HAIRSTON)
Administrative Patent Judge)
)
)
)
) BOARD OF PATENT
JERRY SMITH) APPEALS
Administrative Patent Judge) AND
) INTERFERENCES
)
)
)

LEE E. BARRETT)
Administrative Patent Judge)

Robert D. Marshall, Jr.
TEXAS INSTRUMENTS INCORPORATED
P.O. Box 655474, M/S 219
Dallas, TX 75265